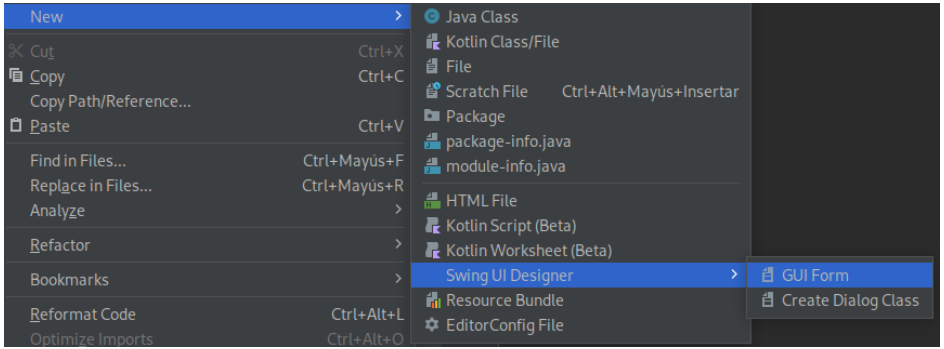
Usando Swing

Componentes JPanel, JLabel, JButton, JSlider
Captura de Eventos

Mas información:
https://www.jetbrains.com/help/idea/gui-designer-basics.html

La interfaz gráfica

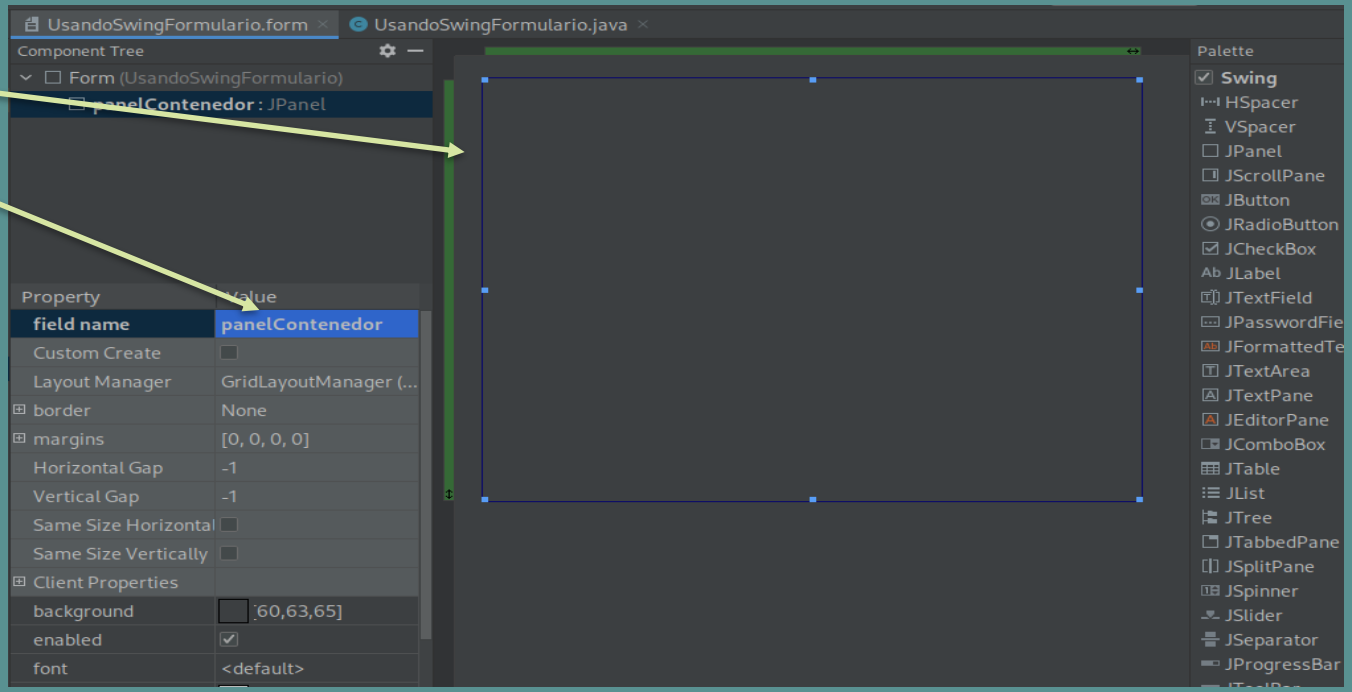
La ventana



- Esta aplicación vamos a realizarla con nuestro entono gráfico.
- Abrimos IntelliJ, y creamos un nuevo formulario de entorno gráfico de usuario (GUI Form) con un nombre adecuado y las opciones por defecto.

- Podemos ver que nos ha generado dos pestañas, una java y otra form. En la form podremos añadir componentes de forma visual desde el panel de elementos que se encuentra a la derecha
- A la izquierda podemos ver la estructura de los componentes y las propiedades del componente seleccionado.
- En la estructura podemos ver que sólo tiene un componente, un JPanel, dentro de él añadiremos los demás componentes.

Le ponemos un nombre al panel




```java
import javax.swing.*;
1 usage
public class UsandoSwingFormulario {
    2 usages
    private JPanel panelContenedor;

    public JPanel getPanelContenedor(){
        return panelContenedor;
    }
}
```

Al ponerle un nombre, se convierte en un atributo en la clase asociada al formulario.

Le creamos un método para poder acceder al panel

La ventana. Ejecutar nuestra aplicación

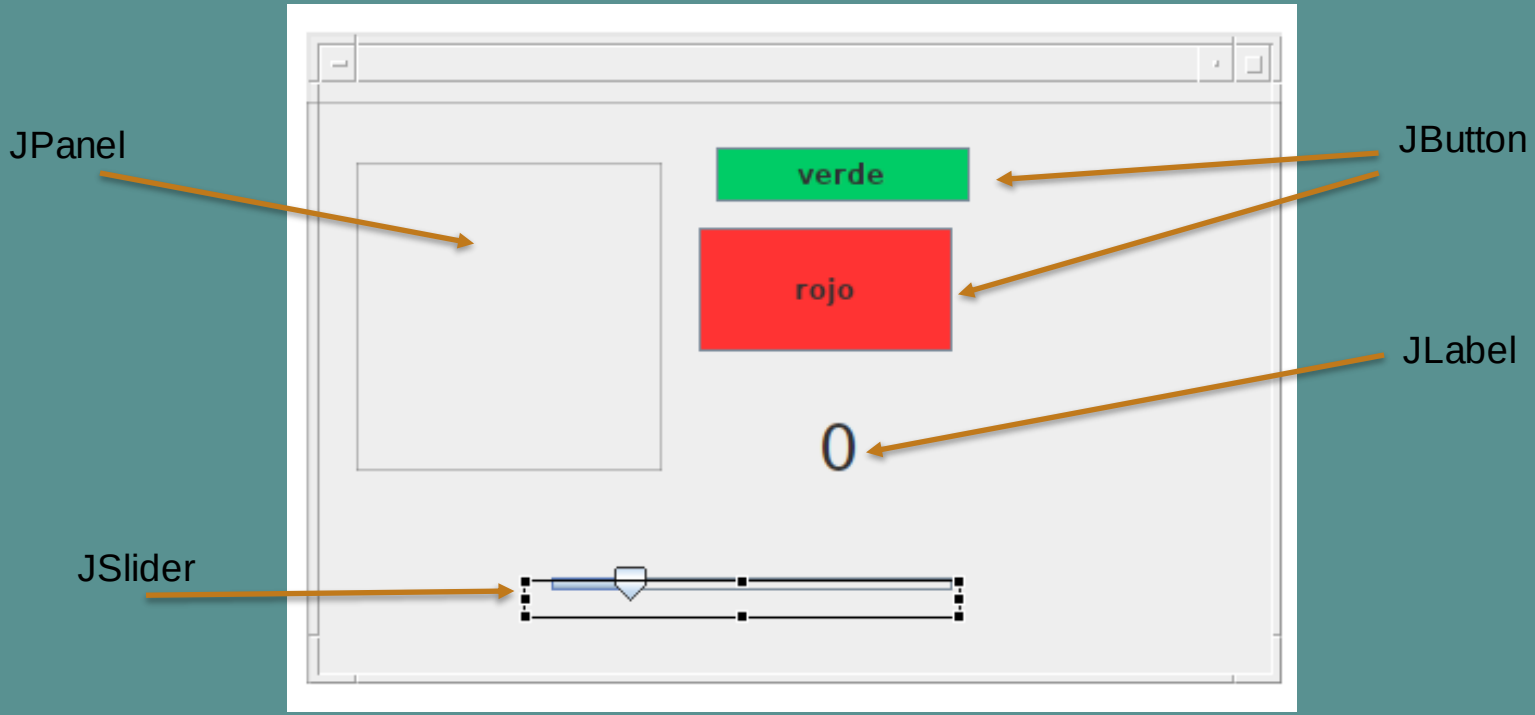
```java
public class UsandoSwing {
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    // Creamos la ventana y le ponemos un titulo
                    JFrame ventana = new JFrame( title: "Usando Swing");
                    // Establecemos la posición y el tamaño de la ventana
                    ventana.setLocation( x: 100, y: 100);
                    ventana.setSize( width: 500, height: 400);

                    // Enlazamos el panel de nuestro formulario con la ventana
                    ventana.setContentPane(new UsandoSwingFormulario().getPanelContenedor());

                    // Establecemos que hacer cuando queremos cerrar la ventana
                    ventana.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
                    // Hacemos visible la ventana
                    ventana.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

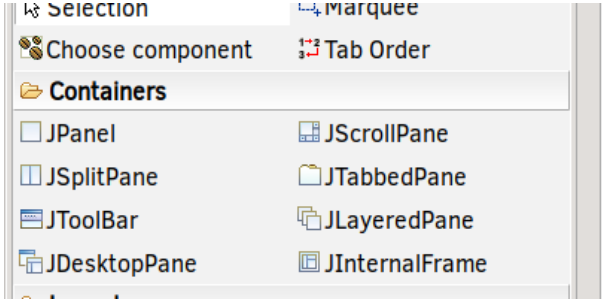
Lo que nos genera IntelliJ no es directamente ejecutable. Para poder lanzar nuestra ventana nos creamos una clase, la clase principal de nuestra aplicación que contendrá el método main.

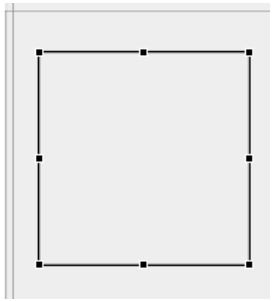
Esta clase crearé la ventana, establecerá valor a sus atributos y enlazará el panel del formulario con el panel de la ventana.


JPanel
JButton
verde
rojo
JLabel
0
JSlider

Componente JPanel

- Este componente es un contenedor. Nos sirve para que añadamos a él otros componente, es una forma de organizarlos.
- En este caso nos va a servir únicamente como un cuadrado que vamos a cambiarle el color
- Elegimos el componente de la paleta, en la sección de contenedores.
- Y lo insertamos en la ventana de nuestra aplicación en la posición y con el tamaño que queramos
- Por defecto el nombre la variable asociada a este componente es panel, la vamos a dejar así





```
JPanel panel = new JPanel();
panel.setBounds(23, 28, 141, 142);
frame.getContentPane().add(panel);
```

El código que se genera.

- Se crea el panel
- Se establece posición y tamaño
- Se añade el componente al panel del frame

Componente Botón

Ahora vamos a insertar dos botones en nuestra aplicación.

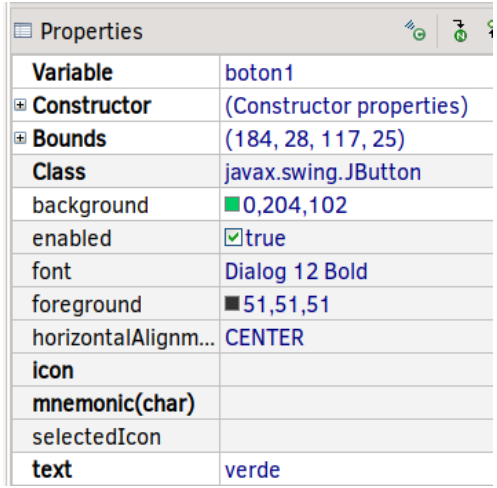
Seleccionamos el componente JButton y a continuación pulsamos sobre una posición en nuestra aplicación. Modificamos su posición y tamaño.

Ahora del primer botón modificamos las siguientes propiedades:

- Variable: boton1
- Text: verde
- Background: Seleccionamos "web safe colors", el verde #00cc66

Del segundo botón:

- Variable: boton2
- Text: rojo
- Background: Seleccionamos "web safe colors", el rojo #ff3333



Componente Botón
Código generado

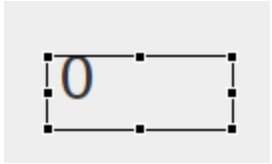
```
JButton boton1 = new JButton("verde");
boton1.setBackground(new Color(0, 204, 102));
boton1.setBounds(184, 28, 117, 25);
frame.getContentPane().add(boton1);

JButton boton2 = new JButton("rojo");
boton2.setBackground(new Color(255, 51, 51));
boton2.setBounds(176, 65, 117, 57);
frame.getContentPane().add(boton2);
```

Destacable tenemos que el texto del botón se pone en el constructor, pero también se podría establecer a través del método setText

Además vemos cómo se establece el color, a través de un valor RGB. El objeto Color también se puede crear con sus valores constantes definidos en la clase Color. Por ejemplo new Color(Color.BLACK)

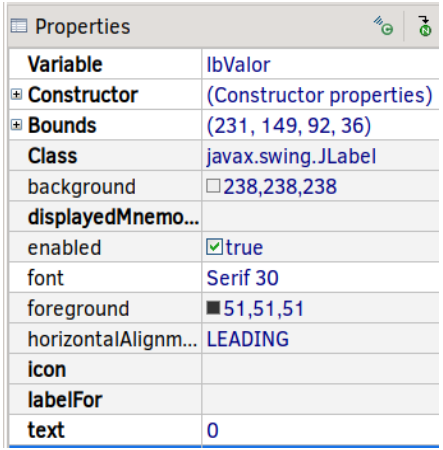
Componente JLabel
Etiquetas



Este componente sirve para mostrar texto o imágenes en nuestras aplicaciones.

Seleccionamos el componente JLabel y creamos una etiqueta en nuestra aplicación, lo seleccionamos y modificamos las siguientes propiedades:

- Variable: lbValor
- Font: Serif, plain, 30
- Text: 0
- Modifica el cuadrado que define la etiqueta para que se vea bien




```
JLabel lbValor = new JLabel("0");
lbValor.setFont(new Font("Serif", Font.PLAIN, 30));
lbValor.setBounds(231, 149, 92, 36);
frame.getContentPane().add(lbValor);
```

Componente JSlider

Este componente sirve para seleccionar un valor de una barra. Podemos establecer entre qué valores estamos seleccionado, y el paso de la selección, entre otras cosas.

Seleccionamos el componente JSlider y situamos el componente en nuestra aplicación. Modificamos las siguientes propiedades:

- Valor máximo: 255

```
JSlider slider = new JSlider();
slider.setMaximum(255);
slider.setBounds(101, 221, 200, 16);
frame.getContentPane().add(slider);
```

Ya hemos definido la forma de la aplicación, ahora vamos a darle vida respondiendo a los

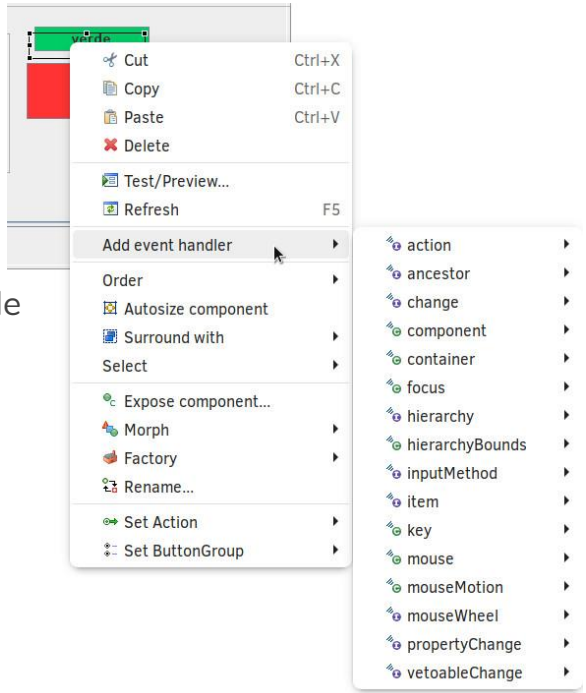
Eventos

¿Qué queremos hacer?

Nuestra aplicación es muy sencilla.

- Al pulsar sobre el botón verde, cambiaremos el fondo del panel.
- Al pulsar sobre el botón rojo, cambiaremos el fondo del panel.
- Cuando movamos el slider, mostraremos el valor en la etiqueta

Pulsaciones de botones ActionEvent



Queremos capturar el evento de acción del botón, que se corresponde con pulsarlo.
Para establecer los eventos en nuestra entorno de desarrollo, seleccionamos el componente y pulsamos con el botón derecho. Seleccionamos "Add event handler", "action", "actionPerformed".

Automáticamente crea el código para responder a ese evento, y sólo nos queda escribir el código de lo que queremos hacer, por ejemplo cambiar el color del panel.

```
boton1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        panel.setBackground(Color.GREEN);
    }
});
```

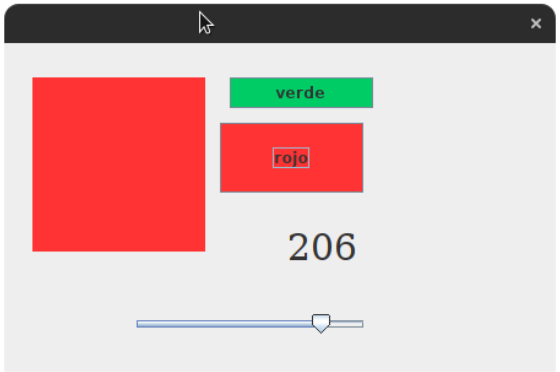
```
boton2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        panel.setBackground(new Color(255,51,51));
    }
});
```

Cambio de estado stateChanged

Ahora queremos capturar el evento de cuando movemos la barra del control deslizante.

Buscamos en evento stateChanged, e introducimos el código para cambiar el valor que nuestra etiqueta con el valor de la barra deslizante.

```
slider.addChangeListener(new ChangeListener() {
    public void stateChanged(ChangeEvent arg0) {
        lbValor.setText(""+slider.getValue());
    }
});
```



Ejercicio 1

Vamos a crear una pequeña aplicación selectora de colores.

Vamos a tener 3 JSlider para seleccionar un valor entre 0 y 255, que corresponderán con los valores RGB de un color.

Mostraremos el valor de los controles de desplazamiento.

Se mostrará el color seleccionado en un componente.

Además mostraremos el valor del color en formato web, que son los valores RGB en hexadecimal precedido por el símbolo #